Table of Contents

The Prototypical Integration of the DES Data Handling System with globusonline.org

Introduction

In March 2011, DES and OSG have agreed to investigate together if the integration of globusonline.org improved the data movement performance of DAF, the DES data handling system. The agreement was limited to an initial phase of exploration of the globusonline.org services, followed by an "integration fest" at Fermilab on Apr 14, 15 2011. OSG User Support was involved for his expertise in the area and because of its existing relationship with the Globus Team.

The current DAF system uses a network of gridftp servers to move data across sites (mainly in TeraGrid and some in Europe) . In short, DAF first verifies that the files at the source (typically the local system) are consistent with the file sizes in the DES replica catalog database. It then transfers the files using some simple static concurrency model, based on experience. It then checks that the sizes of the remote files is the same as in the database, using UberFTP to execute remote "ls" commands.

Two main problems have been identified:

1. The parameters for the transfer are not optimal for transferring a dataset with small and large files
2. The verification step is sequential (ls on one file after another) and is inefficient.

By integrating DAF with globusonline.org, the hope is to address both these issues, relying on the ability of globusonline.org to chose appropriate transfer parameters (concurrency, number of parallel threads, command pipelining, TCP windeos sizes, etc.)

At the end of our investigation, the results are encouraging, even if there is no improvement from our implementation.

Summary

We have integrated DAF with globusonline and run a test with 7,000 files and one with 31,000 (184) GB. We have compared the results of these tests with the current implementation of DAF (using UberFTP).

The time to do transfers + verification of 31,000 files is essentially the same (101 min vs. 103 min.).

Data transfer is faster with globusonline (67 min) than with UberFTP (84 min.) This is a 27% improvement due to a better choice of transfer parameter.

Verification is faster with the current DAF implementation (17 minutes) than with globusonline (35 min). This is because in our implementation, globusonline verifies file sizes doing ls at both src and destination (there is no good interface to do ls of a list of files only at the destination). The uberFTP implementation, instead, verifies file sizes only at the destination and compares the sizes from the db. The UberFTP approach takes 50% less time than our implementation with globusonline: we do not see the effects of command pipelining in data verification (remote ls) with globusonline.

We have listed a set of possible improvements for globusonline (see below). The most important is support for ls of list of files, using command pipelining on the list.

Results

```
- Time to do verification
With globusonline, after transferring the files, we run the transfer a 2nd time to verify that si
globusonline does ls of all files at the src and dest.
We expect that pipelining (multiple gridftp cmd on-the-fly) should be turned on by default.
DAF with !UberFTP does ls ONLY at the dest.

Num Files: 6,978 Time globusonline: 350 sec for remote (src and dest) ls
Time !UberFTP: measured: 223 sec for remote (dest) ls +
```

Possible DES Requests for globusonline

1) Provide an interface to do efficient ls of a list of files (pipelined by globusonline).

2) Better failure model: instead of trying until a deadline, allow retrial of individual transfers. It is not always possible to specify a meaningful deadline. Need more control over retry parameters e.g. number of reties and elapsed time (exponential back off parameters). Example: Std deadline for 10000 files can be 3 hours, but if 9999 are finished after 1 hour, the system should not keep trying to transfer the one failed files for 2 more hours.

3) Add post-transfer verification. Allow specification of synchronization semantics (similarly to the synchronization semantics to decide what files to transfer). For example, we want to request that files be transferred guaranteeing that the size (or CRC, or ...) at src and dest are the same. Globusonline should check this at the end of the transfer.

4) Allow pretend-transfer using a given synchronization policy: this can be used to verify the success of a transfer task (e.g. size is the same OR CRC is the same, etc.) with efficient remote ls (pipelined by globusonline). The pretend-transfer will do the efficient verification without running the transfers.

5) when a file is skipped, the (synchronization) subtask does not show the full path of the skipped file. If a file has a failure on source (e.g. it does not exist), the subtask does not show the path either. Given a list of subtasks with skipped files and failed files at the source, we cannot know which file failed and which file was skipped (...we only have the file names in the text of the event log, that would need parsing).
MORE DETAILS:
When a file is copied, there is a subtask and event log associated -> easy to cheek status
If source = dest, the file is skipped: creates 1 subtask (synchronization subtask) for all skipped files: there is no info on file path.
If source is missing (the file transfer will fail), the subtask does not include the full path of the file: when linking to the relative event log, one needs to parse the event to uniquely identify the file (e.g. the file path is in the text of the error).

Notes from the Planning session

```
Present: Kailash Kotwani, Greg Daues, Marko, Tanya, Gabriele

Date: Apr 14, 2011 and Apr 15, 2011

** Deciding on language: Perl vs. Python vs. cmd line
Prefer REST API because more information and assumed more stable (i.e. no cmd line)
Python for now: fast prototyping; fine moving away from Perl.

** How do we do verify with globusonline for in workflow verify -> transfer -> verify ?
Transfer 100 files twice. The second time, only the failed files will be transferred.
We still need to know if we have failed.

** Security configuration:
```

```
Even if end-point are public, a portal user is going to be given a selection of available endpoin
the portal db controls user access.
If end-point is public, other communities can use them as well.
We should make the end-points private for the portal service certificate.

** Kailash shows us the portal

** Testing method:
We want to
- create a test bed
- try to move files with DAF with the current method
- compare with DAF / globusonline.org

We define 2 endpoints. We define a short dataset for functionality, a long one for performance
(5,000 to 10,000 files for ~50 GB).
Tanya transferred 100 files of different sizes for 10 GB tot: 40 minutes; 2 destinations in the U

** Division of responsibilities
- Kailash integrates Tanya's test globusonline classes with DAF
- Tanya and Marko improve globusonline classes, adding verification logic
- Greg prepares test bed for final verification
```

Current implementation of Tanya's globusonline classes

```
WE DO NOT DO THE FIRST VERIFY OF FILE SIZE FOR COMPARISON WITH THE DB
The globusonline verification implemented in the classes only check that the src and dest sizes
are the same.
This is fine because file sizes at all sites are periodically compared  asynchronously with the D

** Class behavior

- Sunny day scenario:
Initiate transfer; if successful try to transfer again to check the size of the files.
Notes:
Most of the times the sizes will all be the same and no transfer actually happens.
If some sizes are not the same, today the transfer may take as long as a second full deadline
time-window before finishing: the deadline should be changed on the fly proportionally to the num
of files to re-tranfer to avoid this time inefficiency.

- Failure on destination:
Initiate transfer: after deadline if files fail; DO NOT RETRY.
Notes:
WE DO NOT KNOW THAT THE SIZES ARE THE SAME, BECAUSE WE ARE NOT RUNNING
THE TRANSFER AGAIN.
We should try to re-transfer only the successful files

- Failure on source:
Initiate transfer: after deadline, the task fails. DO NOT RETRY
These cannot be distinguished from skipped files (as discussed above).
```

-- GabrieleGarzoglio - 18 Apr 2011

This topic: Engagement > DESIntegrationWithGlobusonline
Topic revision: r1 - 18 Apr 2011 - 18:53:24 - GabrieleGarzoglio

MyOSG | BDII | GOC Ticket | OIM | OSG Software Cache | TWiki | GOC News Blog | Report Bugs